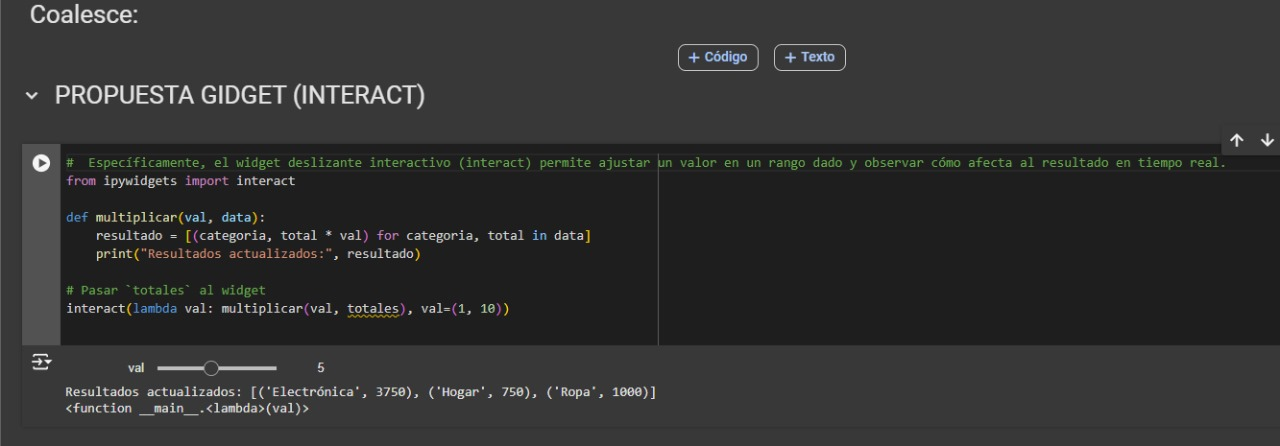
**Actividad 14 - 2**

**Propuestas para la mejora de la actividad 13**

**Propuesta 1: Gidget (INTERACT)**

Este código básicamente te deja interactuar con un control deslizante, como un slider, para ajustar un número que luego se usa para multiplicar los valores de un conjunto de datos. Este conjunto tiene categorías, como "Electrónica" o "Ropa", con sus valores, y lo que pasa es que cada vez que mueves el slider, se recalculan esos valores multiplicándolos por el número que elegiste.

Lo interesante es que puedes ver los resultados inmediatamente en la consola, o sea, todo se actualiza al momento. Es una forma súper útil si quieres analizar cómo un cambio en un número afecta a los datos, pero sin hacerlo manualmente cada vez. Todo es automático y directo.

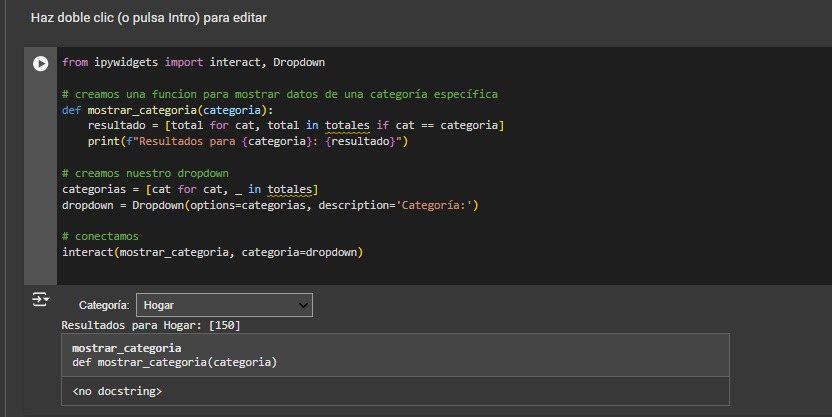
**Propuesta 2: Agregar gráficos para representar resultados**

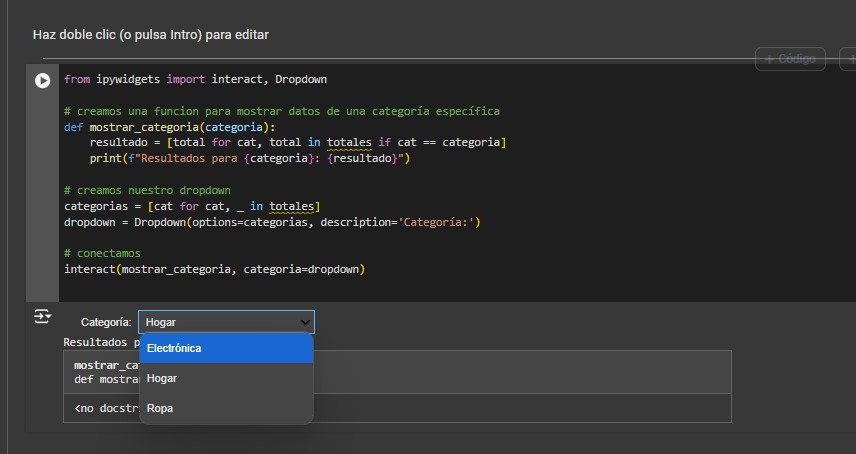
****

Aquí lo que se está haciendo es generar un gráfico de barras para visualizar datos, específicamente los totales por categoría. Básicamente, toma una lista de categorías junto con sus valores (en este caso parece ser algo como "Electrónica", "Hogar", "Ropa") y los representa gráficamente.

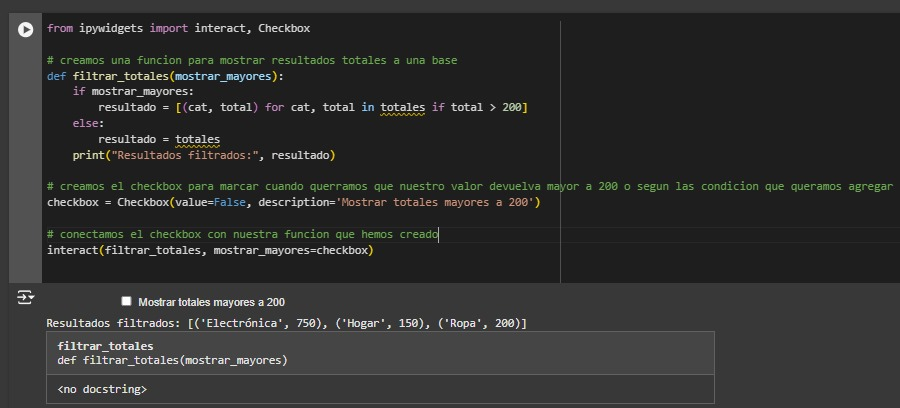
La idea es que cada categoría tenga un color diferente (como rojo, verde y azul), y el gráfico tiene un título bastante claro, "Totales por Categoría". También se añaden etiquetas en los ejes: uno muestra las categorías y el otro los totales. Incluso hay una cuadrícula en el fondo para que sea más fácil leer los valores.

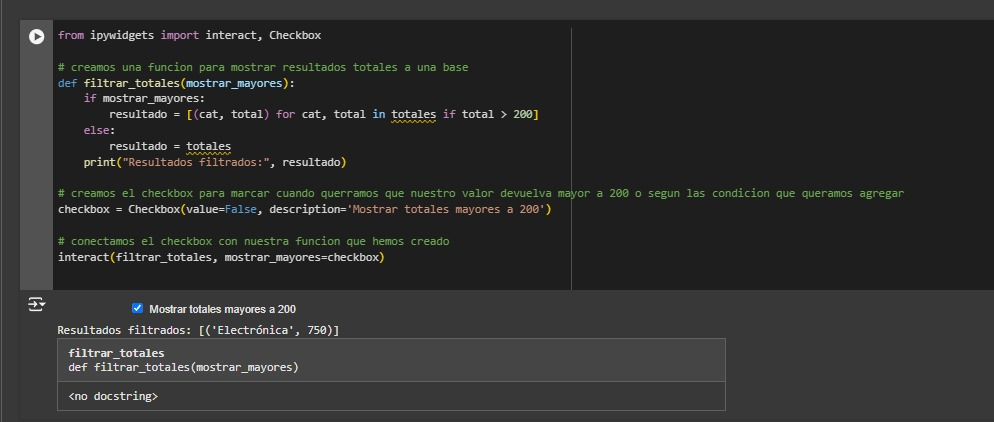
Al final, se muestra el gráfico. En resumen, esto es una forma bastante atractiva de representar datos y hacerlos más visuales en lugar de solo mostrarlos como texto o números. Es una manera genial de interpretar rápidamente las diferencias entre categorías.

**Propuesta 3: Añadir un cuadro que nos permita seleccionar una categoría especifica y mostrar sus datos:**



Este sistema interactivo permite analizar fácilmente datos categorizados seleccionando opciones de un menú desplegable. Si se escala con Apache Spark, podríamos manejar millones de categorías y sus valores, procesándolos en tiempo real y proporcionando los mismos resultados de manera rápida y eficiente.

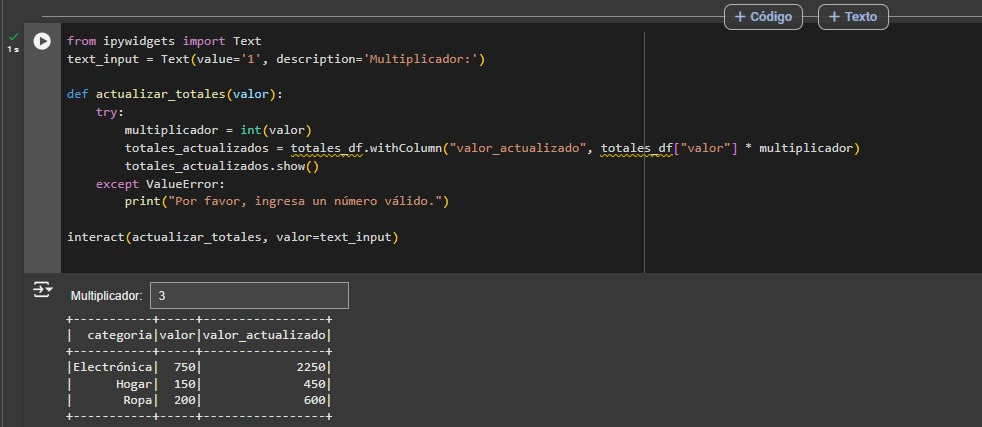
**Propuesta 4: Añadimos un cuadro que lo podremos marcar (checkBox) para cuando queramos que nos muestre una condición en especifico**

****

Este código es útil para filtrar dinámicamente datos y ajustar vistas según criterios específicos (en este caso, valores mayores a 200).

Permite explorar los datos de forma interactiva sin necesidad de modificar el código manualmente

**Propuesta 5: Creamos un cuadro txt donde se pueda ingresar un valor del número multiplicador**

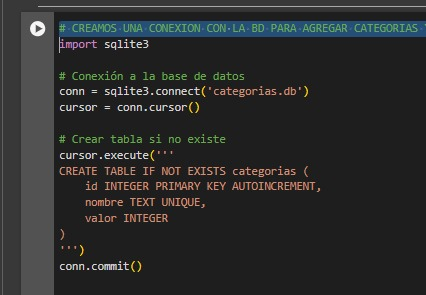


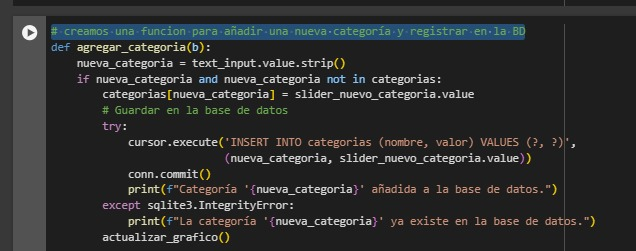
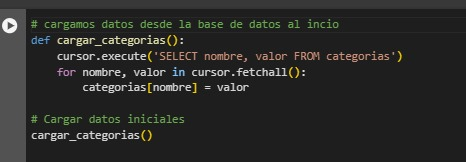
Análisis de Escenarios: Este sistema es útil para analizar cómo un cambio en un multiplicador afecta a los datos, como calcular ingresos proyectados o costos en diferentes escenarios.

Aplicación en Big Data: Usando Spark, podrías aplicar este enfoque para personalizar cálculos en datasets masivos, como ventas globales o inventarios.

Interactividad Dinámica: Los usuarios pueden experimentar con diferentes multiplicadores y ver los resultados instantáneamente.

* **Creamos una conexión con la bd para agregar categorías y que nos muestre una tabla analítica para ver cómo se manejan los resultados de entrar que colocamos a nuestras categorías**

****

* **Creamos una función para añadir una nueva categoría y registrar en la BD**
* **Ahora cargaremos datos desde la base de datos al inicio para hacer nuestra prueba**

**Propuesta General: Sistema Interactivo de Gestión y Visualización de Categorías con Conexión a Base de Datos**

from ipywidgets import interact, Text, IntSlider, VBox, HBox, Button, Output

import matplotlib.pyplot as plt

import sqlite3

# podemos hacer la carga desde nuestra BD ( podemos hacer la conexion con una BD que ya tenemos y solo vincularla especificando que BD queremos trabajar)

# hacemos el llamdo a nuestra conexion con nuestra BD

conn = sqlite3.connect('categorias.db')

cursor = conn.cursor()

# creamos la tabla si no existe

cursor.execute('''

CREATE TABLE IF NOT EXISTS categorias (

id INTEGER PRIMARY KEY AUTOINCREMENT,

nombre TEXT UNIQUE,

valor INTEGER

)

''')

conn.commit()

# manejamos los datos iniciales

categorias = {}

output = Output()

def cargar\_categorias():

cursor.execute('SELECT nombre, valor FROM categorias')

for nombre, valor in cursor.fetchall():

categorias[nombre] = valor

cargar\_categorias()

# Creamos una funcion para actualizar nuestro grafico que nos ayudara en el analisis

def actualizar\_grafico():

output.clear\_output()

with output:

plt.bar(categorias.keys(), categorias.values(), color=['#FF5733', '#33FF57', '#3357FF', '#FFD700'])

plt.title("Totales por Categoría", fontsize=16, fontweight='bold')

plt.xlabel("Categorías")

plt.ylabel("Totales")

plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.show()

# ahora una funcion def para nueva categoria

def agregar\_categoria(b):

nueva\_categoria = text\_input.value.strip()

if nueva\_categoria and nueva\_categoria not in categorias:

categorias[nueva\_categoria] = slider\_nuevo\_categoria.value

try:

cursor.execute('INSERT INTO categorias (nombre, valor) VALUES (?, ?)',

(nueva\_categoria, slider\_nuevo\_categoria.value))

conn.commit()

print(f"Categoría '{nueva\_categoria}' añadida a la base de datos.")

except sqlite3.IntegrityError:

print(f"La categoría '{nueva\_categoria}' ya existe en la base de datos.")

actualizar\_grafico()

# aqui decidimos crear un slider para manipular los valores iniciales de nuestras categorias a agregar

slider\_nuevo\_categoria = IntSlider(value=100, min=1, max=500, description='Valor Inicial:')

# creamos un txt input para mandar el nombre de nuestra categoria

text\_input = Text(value='', placeholder='Nueva Categoría', description='Categoría:')

# nuestro boton para añadir

boton\_agregar = Button(description="Añadir Categoría", button\_style="success")

boton\_agregar.on\_click(agregar\_categoria)

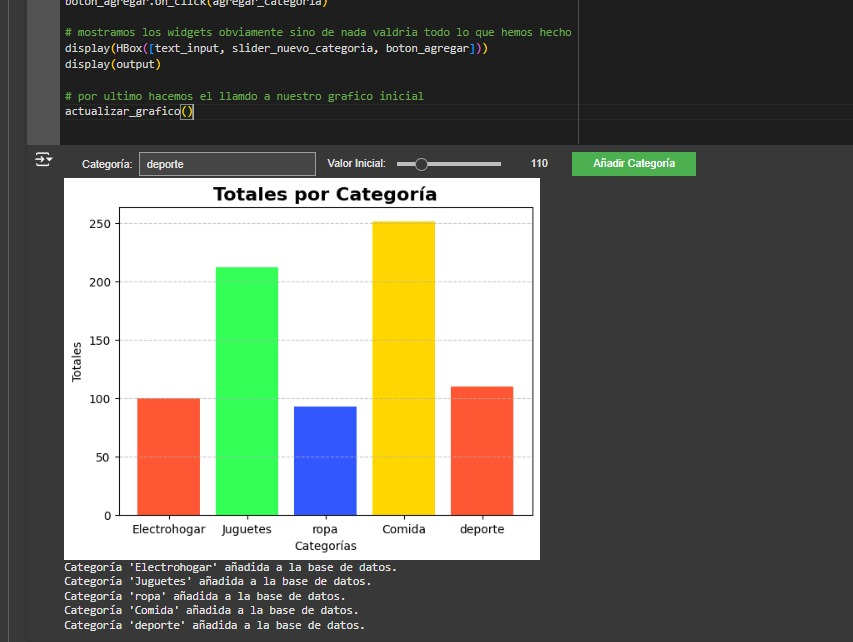
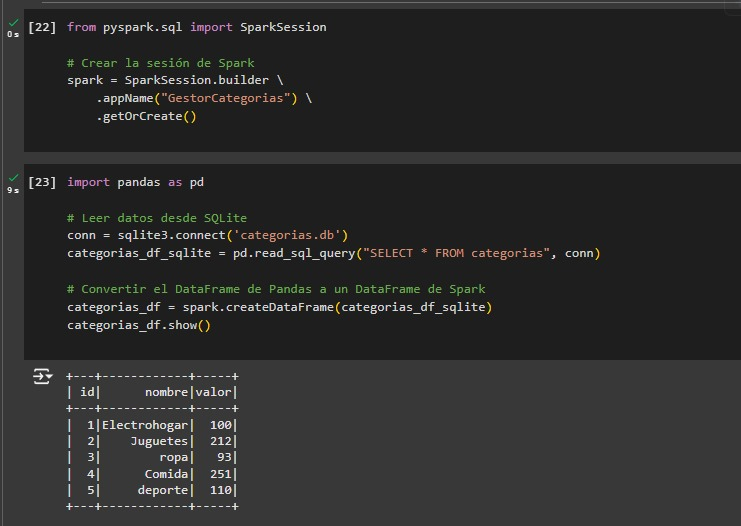
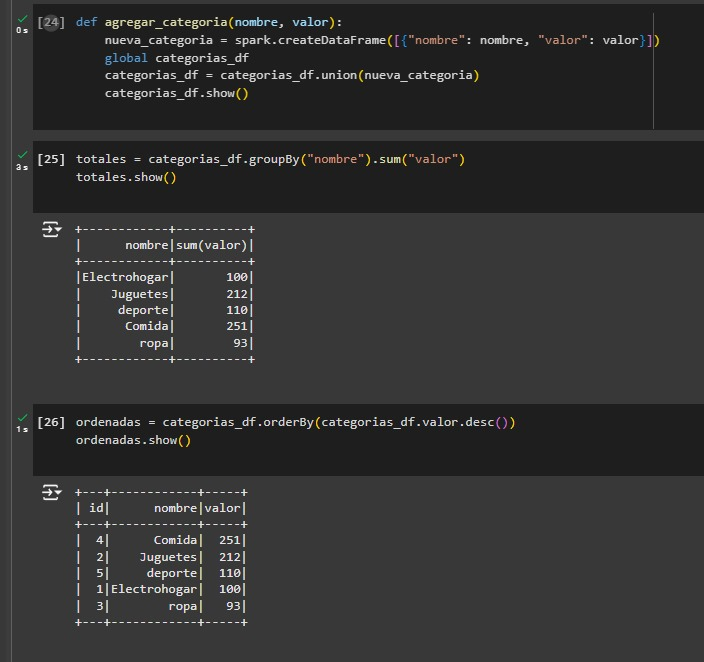
# mostramos los widgets obviamente sino de nada valdria todo lo que hemos hecho

display(HBox([text\_input, slider\_nuevo\_categoria, boton\_agregar]))

display(output)

# por ultimo hacemos el llamdo a nuestro grafico inicial

actualizar\_grafico()

* ****Aquí hemos comprobado que se hizo el agregar en nuestra BD
* Si integramos Apache Spark con este sistema es una idea excelente para escalarlo y manejar grandes volúmenes de datos. Aquí explicamos cómo lo haría paso a paso, utilizando Spark DataFrames y RDDs como base para procesar la información.